Tightness and Computation Assessment of Worst-Case Delay Bounds in Wormhole Networks-On-Chip

Frédéric Giroudot, Ahlem Mifdaoui
ISAE Supaéro – Université de Toulouse

November 6, 2019

RTNS 2019

Outline

Outline

Problem Statement

BATA

Tightness Analysis

Computation Analysis

Case Study

Network-on-chip (NoC)

- Widely used and scalable interconnect for manycore platforms and SoCs

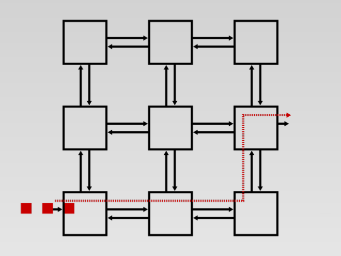
Network-on-chip (NoC)

- Widely used and scalable interconnect for manycore platforms and SoCs
- Low latency, high throughput

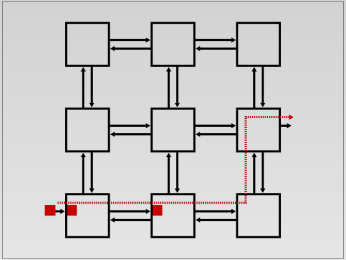
Network-on-chip (NoC)

- Widely used and scalable interconnect for manycore platforms and SoCs
- Low latency, high throughput
- Wormhole Routing allows small buffer sizes

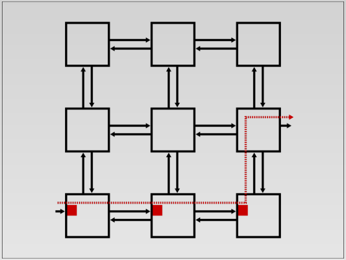
Wormhole Routing Intuition



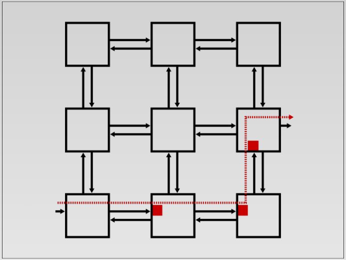
Wormhole Routing Intuition



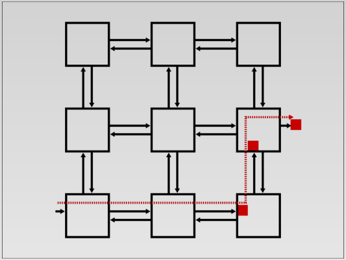
Wormhole Routing Intuition



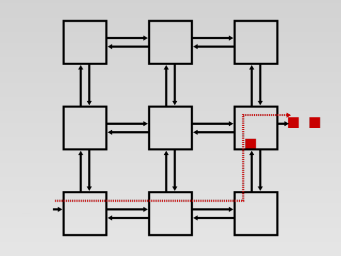
Wormhole Routing Intuition



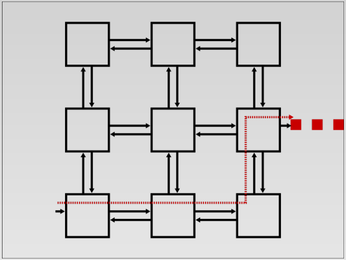
Wormhole Routing Intuition



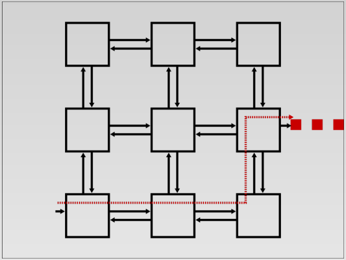
Wormhole Routing Intuition



Wormhole Routing Intuition

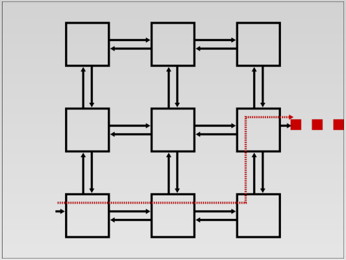


Wormhole Routing Intuition

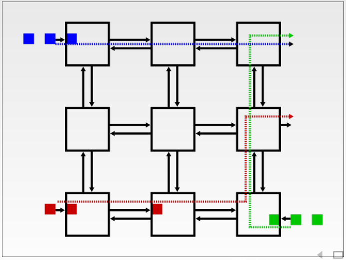


Now with congestion :

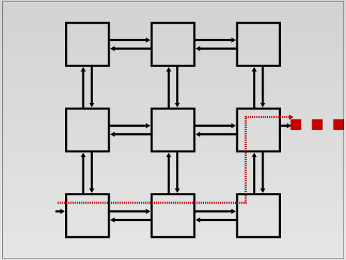
Wormhole Routing Intuition



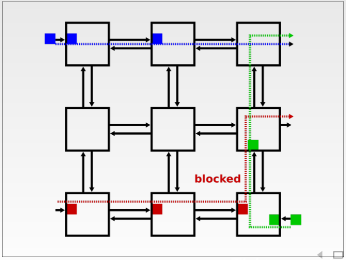
Now with congestion :



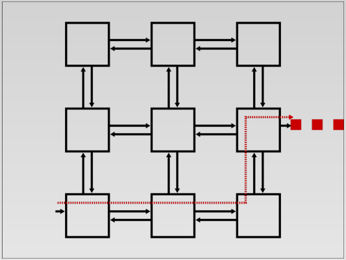
Wormhole Routing Intuition



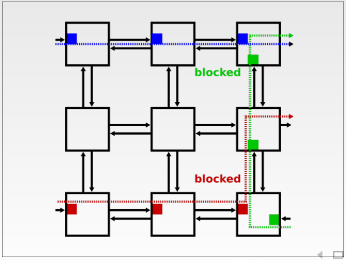
Now with congestion :



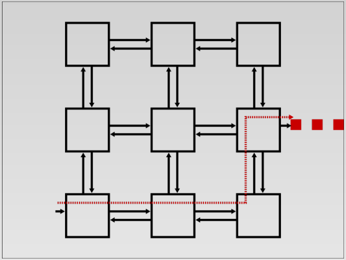
Wormhole Routing Intuition



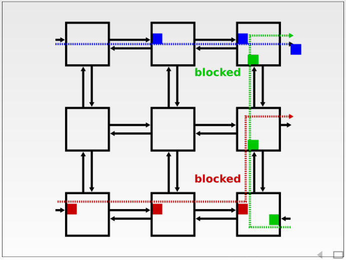
Now with congestion :



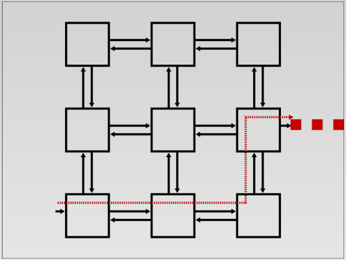
Wormhole Routing Intuition



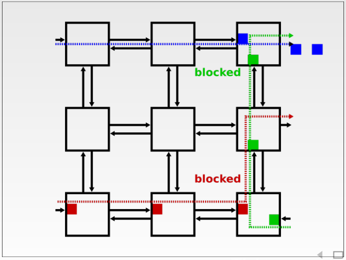
Now with congestion :



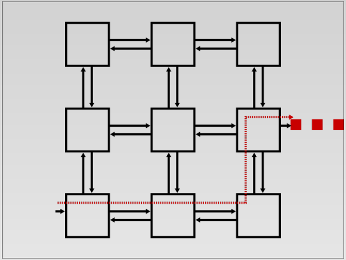
Wormhole Routing Intuition



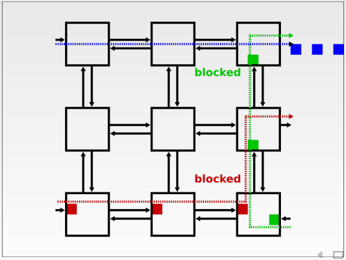
Now with congestion :



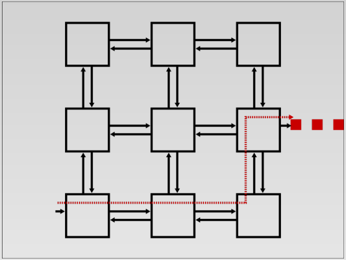
Wormhole Routing Intuition



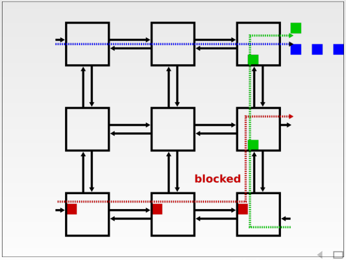
Now with congestion :



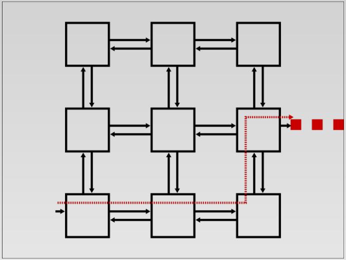
Wormhole Routing Intuition



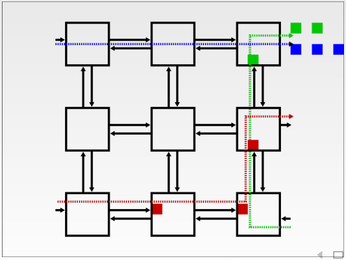
Now with congestion :



Wormhole Routing Intuition



Now with congestion :



Challenges of wormhole

- Packets are spread in the network when congestion occurs

Challenges of wormhole

- Packets are spread in the network when congestion occurs
- Lossless transmission creates **backpressure**

Challenges of wormhole

- Packets are spread in the network when congestion occurs
- Lossless transmission creates **backpressure**
- Flows that do not share resources may interfere (**indirect blocking**)

Main existing approaches

- Scheduling Theory (ST): University of York [1], HUST and KTH [2], Mälardalen University [3]

Main existing approaches

- Scheduling Theory (ST): University of York [1], HUST and KTH [2], Mälardalen University [3]
- Compositional Performance Analysis (CPA): TU Braunschweig [4, 5]

Main existing approaches

- Scheduling Theory (ST): University of York [1], HUST and KTH [2], Mälardalen University [3]
- Compositional Performance Analysis (CPA): TU Braunschweig [4, 5]
- Recursive Calculus (RC): ISAE [6], ENSEEIHT [7]

Main existing approaches

- Scheduling Theory (ST): University of York [1], HUST and KTH [2], Mälardalen University [3]
- Compositional Performance Analysis (CPA): TU Braunschweig [4, 5]
- Recursive Calculus (RC): ISAE [6], ENSEEIHT [7]
- **Network Calculus**: KTH [8, 9, 10], **our approach BATA [11]**

Approach	ST				CPA		RC			NC			
Contribution	[2]	[1]	[3]	[12]	[4]	[5]	[6]	[13]	[7]	[8]	[9]	[14]	**BATA[11]**
wormhole	x	x	x	x	x	x	x	x	x	x		x	**x**
multiple VCs	x	x	x	x		x					x	x	**x**
priority sharing		x	x		x	x				x	x		**x**
VCs sharing						x							**x**
flows serialization				x		x			x		x	x	**x**
$B = 1$ flit	x		x	x				x	x	x			**x**
$L \leq B$	x			x	x	x		x		x	x		**x**
$B \leq L$	x			x				x		x		x	**x**

Approach	ST				CPA		RC			NC			
Contribution	[2]	[1]	[3]	[12]	[4]	[5]	[6]	[13]	[7]	[8]	[9]	[14]	**BATA[11]**
wormhole	x	x	x	x	x	x	x	x	x	x		x	**x**
multiple VCs	x	x	x	x		x					x	x	**x**
priority sharing		x	x		x	x				x	x		**x**
VCs sharing						x							**x**
flows serialization				x		x			x		x	x	**x**
$B = 1$ flit	x		x	x				x	x	x			**x**
$L \leq B$	x			x	x	x		x		x	x		**x**
$B \leq L$	x			x				x		x		x	**x**

We presented BATA [11] (Buffer-Aware Timing Analysis)
Based on **Network Calculus**:

Approach	ST				CPA		RC			NC			
Contribution	[2]	[1]	[3]	[12]	[4]	[5]	[6]	[13]	[7]	[8]	[9]	[14]	**BATA[11]**
wormhole	x	x	x	x	x	x	x	x	x	x		x	**x**
multiple VCs	x	x	x	x		x					x	x	**x**
priority sharing		x	x		x	x				x	x		**x**
VCs sharing						x							**x**
flows serialization				x		x			x		x	x	**x**
$B = 1$ flit	x		x	x				x	x	x			**x**
$L \leq B$	x			x	x	x		x		x	x		**x**
$B \leq L$	x			x				x		x		x	**x**

We presented BATA [11] (Buffer-Aware Timing Analysis)
Based on **Network Calculus**:

- address limitations of existing approaches

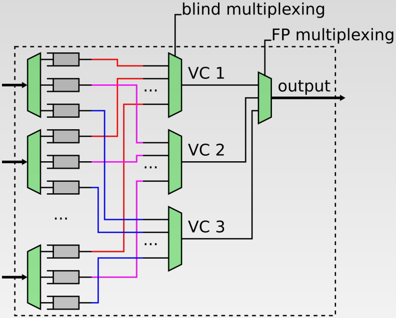
Approach	ST				CPA		RC			NC			
Contribution	[2]	[1]	[3]	[12]	[4]	[5]	[6]	[13]	[7]	[8]	[9]	[14]	**BATA[11]**
wormhole	x	x	x	x	x	x	x	x	x	x		x	**x**
multiple VCs	x	x	x	x		x					x	x	**x**
priority sharing		x	x		x	x				x	x		**x**
VCs sharing						x							**x**
flows serialization				x		x			x		x	x	**x**
$B = 1$ flit	x		x	x				x	x	x			**x**
$L \leq B$	x			x	x	x		x		x	x		**x**
$B \leq L$	x			x				x		x		x	**x**

We presented BATA [11] (Buffer-Aware Timing Analysis)
Based on **Network Calculus**:

- address limitations of existing approaches
- provide a general and tight approach

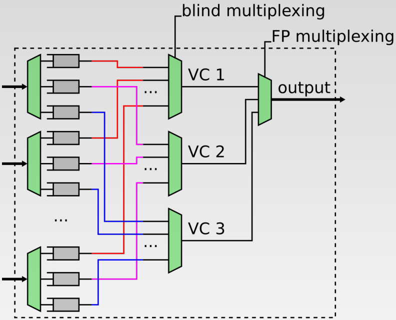
Outline

System Model and Assumptions



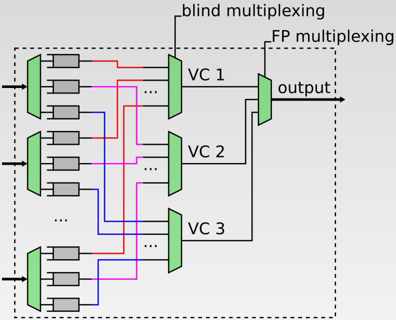
- **Input-buffered routers** (minor changes for output-buffered routers)

System Model and Assumptions



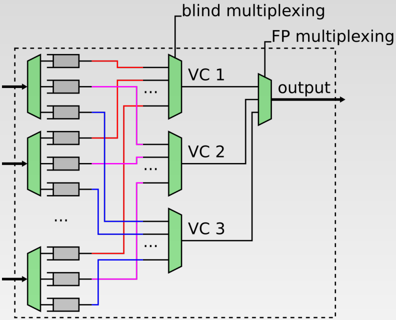
- **Input-buffered routers** (minor changes for output-buffered routers)
- **Arbitrary multiplexing** within a VC

System Model and Assumptions



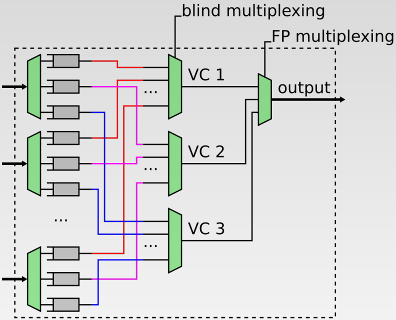
- **Input-buffered routers** (minor changes for output-buffered routers)
- **Arbitrary multiplexing** within a VC
- **Priority-based** arbitration between VCs

System Model and Assumptions



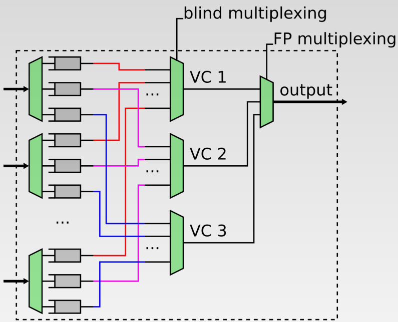
- **Input-buffered routers** (minor changes for output-buffered routers)
- **Arbitrary multiplexing** within a VC
- **Priority-based** arbitration between VCs
- **VC-sharing**

System Model and Assumptions



- **Input-buffered routers** (minor changes for output-buffered routers)
- **Arbitrary multiplexing** within a VC
- **Priority-based** arbitration between VCs
- **VC-sharing**
- **Priority sharing**

System Model and Assumptions



- **Input-buffered routers** (minor changes for output-buffered routers)
- **Arbitrary multiplexing** within a VC
- **Priority-based** arbitration between VCs
- **VC-sharing**
- **Priority sharing**
- **Flit-level preemption**

BATA Methodology

Main idea : compute the **end-to-end service curve** for a *foi f* :

$$\beta_f(t) = R_f\,(t - T_f\ )^+$$

BATA Methodology

Main idea : compute the **end-to-end service curve** for a *foi* f :

$$\beta_f(t) = R_f\,(t - T_f\,)^+$$

where :

- R_f : bottleneck rate along the flow path
- T_f : service latency

BATA Methodology

Main idea : compute the **end-to-end service curve** for a *foi* f :

$$\beta_f(t) = R_f\,(t - T_f\,)^+$$

where :

- R_f : bottleneck rate along the flow path
- T_f : service latency

$$T_f = T_{DB} + T_{IB} + T_{\mathbb{P}_f}$$

BATA Methodology

Main idea : compute the **end-to-end service curve** for a *foi* f :

$$\beta_f(t) = R_f\,(t - T_f\,)^+$$

where :

- R_f : bottleneck rate along the flow path
- T_f : service latency

$$T_f = T_{DB} + T_{IB} + T_{\mathbb{P}_f}$$

Direct blocking latency, due to flows sharing resources with f

BATA Methodology

Main idea : compute the **end-to-end service curve** for a *foi f* :

$$\beta_f(t) = R_f\,(t - T_f\,)^+$$

where :

- R_f : bottleneck rate along the flow path
- T_f : service latency

$$T_f = T_{DB} + T_{IB} + T_{\mathbb{P}_f}$$

Technological latency (routers)

BATA Methodology

Main idea : compute the **end-to-end service curve** for a *foi f* :

$$\beta_f(t) = R_f\,(t - T_f\,)^+$$

where :

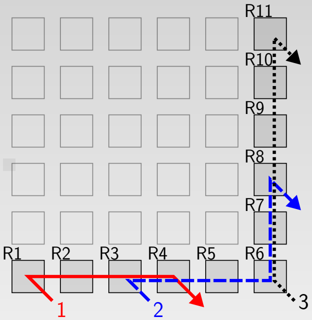
- R_f : bottleneck rate along the flow path
- T_f : service latency

$$T_f = T_{DB} + T_{IB} + T_{\mathbb{P}_f}$$

Indirect blocking latency, due to backpressure: takes buffer size into account

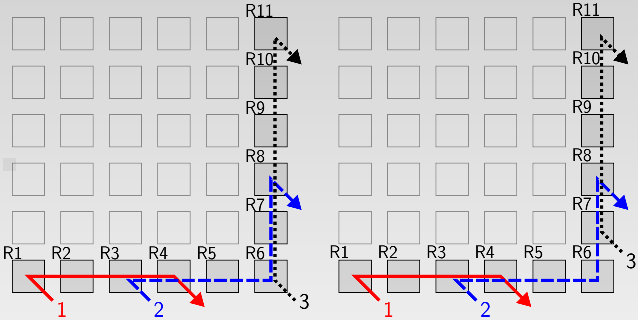
Buffer size matters!

Two slightly different configurations:



Buffer size matters!

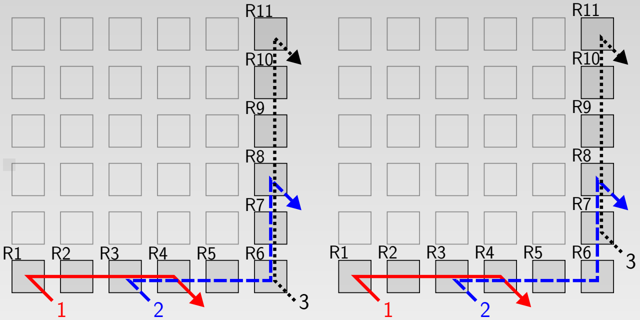
Two slightly different configurations:



Flow 2 shares resources with flows 1 and 3 $\Rightarrow$ indirect blocking

Buffer size matters!

Two slightly different configurations:



Flow 2 shares resources with flows 1 and 3 $\Rightarrow$ indirect blocking

. . . Really?

Buffer size matters!

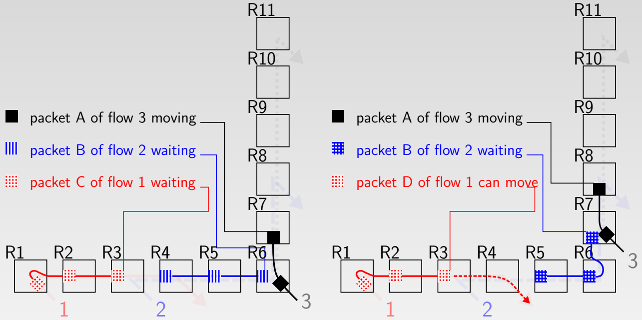
Two slightly different configurations:

Let's see how packets queue!

Buffer size matters!

Two slightly different configurations:

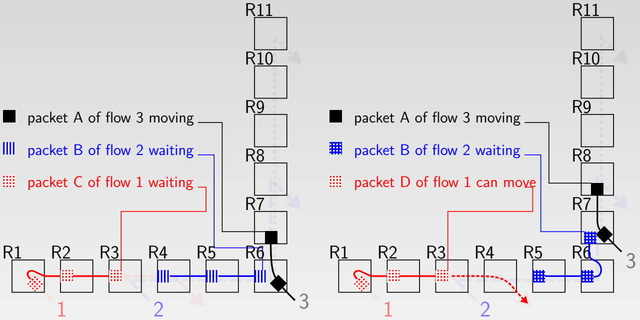
Let's see how packets queue!



Buffer size matters!

Two slightly different configurations:

Let's see how packets queue!



Flow 3 blocks flow 1 in only one configuration!

Outline

To assess tightness, we conduct a sensitivity analysis:

To assess tightness, we conduct a sensitivity analysis:

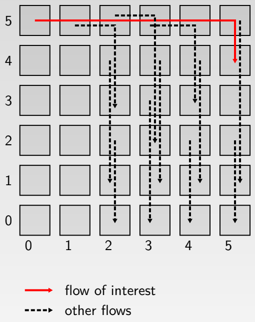
- Determine how the input parameters (buffer size, packet length and flow rate) impact the computed delay bound

To assess tightness, we conduct a sensitivity analysis:

- Determine how the input parameters (buffer size, packet length and flow rate) impact the computed delay bound
- Use the insight to study tightness

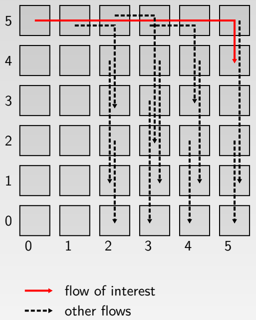
To assess tightness, we conduct a sensitivity analysis:

- Determine how the input parameters (buffer size, packet length and flow rate) impact the computed delay bound
- Use the insight to study tightness



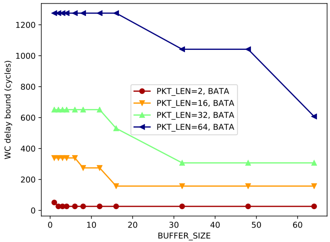
To assess tightness, we conduct a sensitivity analysis:

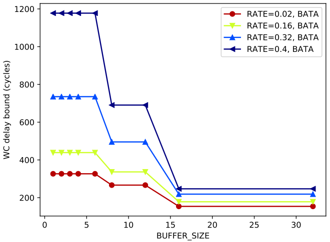
- Determine how the input parameters (buffer size, packet length and flow rate) impact the computed delay bound
- Use the insight to study tightness



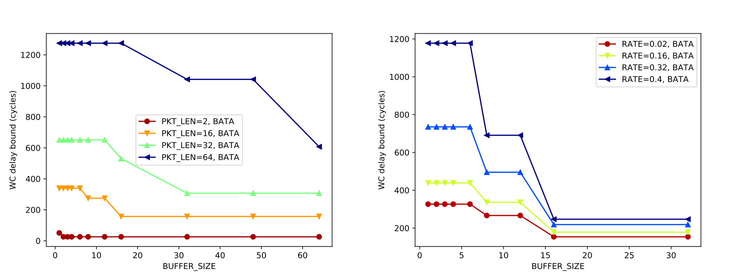
We use a configuration exhibiting a lot of potential interference between flows

Results


WC delay bound (cycles)
BUFFER_SIZE
PKT_LEN=2, BATA
PKT_LEN=16, BATA
PKT_LEN=32, BATA
PKT_LEN=64, BATA
0
200
400
600
800
1000
1200
0
10
20
30
40
50
60

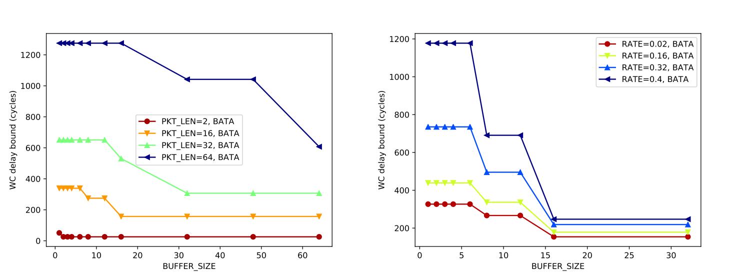

WC delay bound (cycles)
BUFFER_SIZE
RATE=0.02, BATA
RATE=0.16, BATA
RATE=0.32, BATA
RATE=0.4, BATA
200
400
600
800
1000
1200
0
5
10
15
20
25
30

Results



"Step decrease" of delay bounds when $B \nearrow$

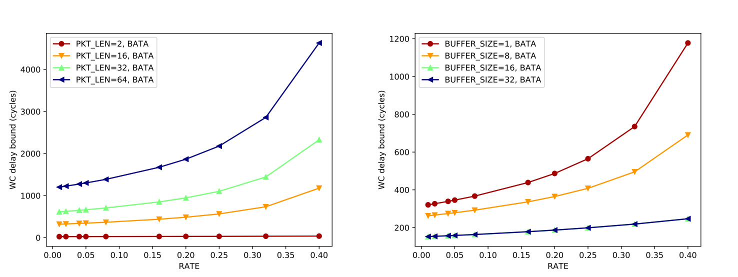
Results



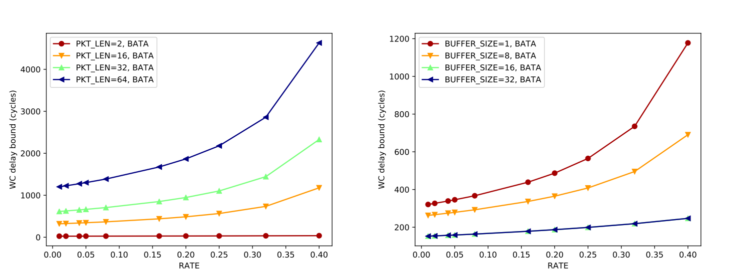
"Step decrease" of delay bounds when $B \nearrow$

Buffer size: directly impacts spread index and complexity of IB patterns

Results

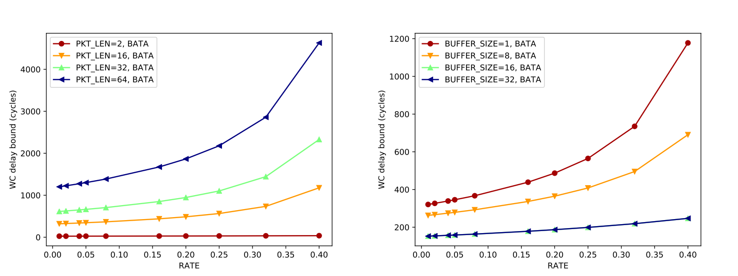

PKT_LEN=2, BATA
PKT_LEN=16, BATA
PKT_LEN=32, BATA
PKT_LEN=64, BATA
WC delay bound (cycles)
RATE
0
1000
2000
3000
4000
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
BUFFER_SIZE=1, BATA
BUFFER_SIZE=8, BATA
BUFFER_SIZE=16, BATA
BUFFER_SIZE=32, BATA
200
400
600
800
1000
1200

Results



Increase of delay bounds with rate (polynomial)

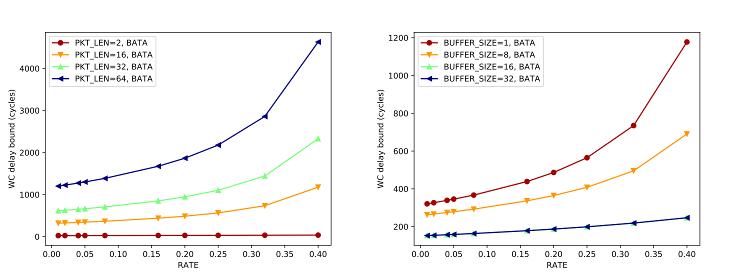
Results



Increase of delay bounds with rate (polynomial)

Rate: terms in ρT in delay bound computation due to burst propagation

Results



Increase of delay bounds with rate (polynomial)

Rate: terms in ρT in delay bound computation due to burst propagation

We perform the tightness analysis while varying **buffer size** and **flow rate**

Tightness Analysis

Simulate the configuration with Noxim [15] while varying the most sensitive parameters

Tightness Analysis

Simulate the configuration with Noxim [15] while varying the most sensitive parameters

Compute **tightness ratio** $\tau = \frac{\text{Simulated worst-case}}{\text{Theoretical bound}}$

Tightness Analysis

Simulate the configuration with Noxim [15] while varying the most sensitive parameters

Compute **tightness ratio** $\tau = \frac{\text{Simulated worst-case}}{\text{Theoretical bound}}$

$\rightarrow$ **lower bound** of the actual tightness!

Tightness Analysis

Simulate the configuration with Noxim [15] while varying the most sensitive parameters

Compute **tightness ratio** $\tau = \frac{\text{Simulated worst-case}}{\text{Theoretical bound}}$

$\rightarrow$ **lower bound** of the actual tightness!

Rate	8%			32%		
Buffer	4	8	16	4	8	16
Tightness Statistics						
Average	**70.1%**	**72.1%**	**80.8%**	**49.7%**	**64.2%**	**79.8%**
Max	**91.7%**	**92.0%**	**88.3%**	**95.6%**	**88.9%**	**97.3%**
Min	**40.6%**	**38.1%**	**48.9%**	**20.8%**	**33.3%**	**43.8%**

Tightness Analysis

Simulate the configuration with Noxim [15] while varying the most sensitive parameters

Compute **tightness ratio** $\tau = \frac{\text{Simulated worst-case}}{\text{Theoretical bound}}$

$\rightarrow$ **lower bound** of the actual tightness!

Rate	8%			32%		
Buffer	4	8	16	4	8	16
Tightness Statistics						
Average	**70.1%**	**72.1%**	**80.8%**	**49.7%**	**64.2%**	**79.8%**
Max	**91.7%**	**92.0%**	**88.3%**	**95.6%**	**88.9%**	**97.3%**
Min	**40.6%**	**38.1%**	**48.9%**	**20.8%**	**33.3%**	**43.8%**

Tightness Analysis

Simulate the configuration with Noxim [15] while varying the most sensitive parameters

Compute **tightness ratio** $\tau = \frac{\text{Simulated worst-case}}{\text{Theoretical bound}}$

$\rightarrow$ **lower bound** of the actual tightness!

Rate	8%			32%		
Buffer	4	8	16	4	8	16
Tightness Statistics						
Average	**70.1%**	**72.1%**	**80.8%**	**49.7%**	**64.2%**	**79.8%**
Max	**91.7%**	**92.0%**	**88.3%**	**95.6%**	**88.9%**	**97.3%**
Min	**40.6%**	**38.1%**	**48.9%**	**20.8%**	**33.3%**	**43.8%**

Results

- Average tightness up to 80%

Tightness Analysis

Simulate the configuration with Noxim [15] while varying the most sensitive parameters

Compute **tightness ratio** $\tau = \frac{\text{Simulated worst-case}}{\text{Theoretical bound}}$

$\rightarrow$ **lower bound** of the actual tightness!

Rate	8%			32%		
Buffer	4	8	16	4	8	16
Tightness Statistics						
Average	**70.1%**	**72.1%**	**80.8%**	**49.7%**	**64.2%**	**79.8%**
Max	**91.7%**	**92.0%**	**88.3%**	**95.6%**	**88.9%**	**97.3%**
Min	**40.6%**	**38.1%**	**48.9%**	**20.8%**	**33.3%**	**43.8%**

Results

- Average tightness up to 80%
- Tightness is better for large buffers and small rates

Outline

Computational Analysis

Goal

Determine how well the approach scales

Computational Analysis

Goal

Determine how well the approach scales

Method

- Use many randomly generated configurations with different numbers of flows (4 to 128)

Computational Analysis

Goal

Determine how well the approach scales

Method

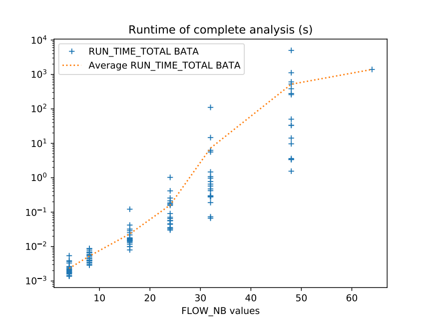
- Use many randomly generated configurations with different numbers of flows (4 to 128)
- Perform timing analysis and measure the computation times (total, IB analysis, service curve computation)

Results

Complete analysis

Results

Complete analysis

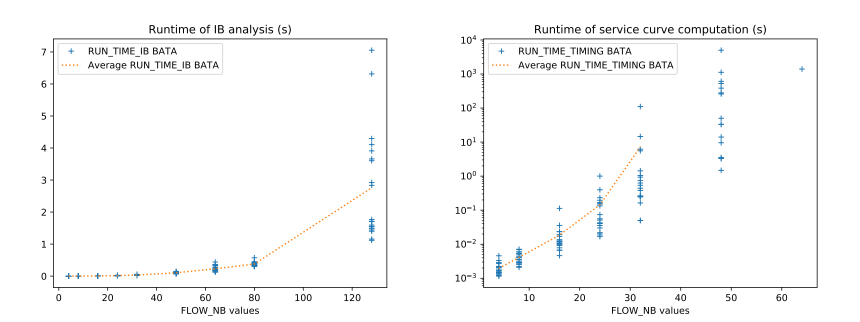


Results

Split IB analysis and service curve computation

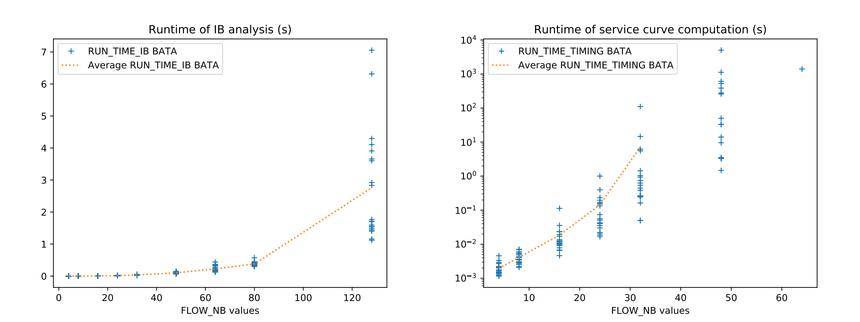
Results

Split IB analysis and service curve computation



Results

Split IB analysis and service curve computation



Complexity: **service curve computation**

Outline

Case Study

Control of an autonomous vehicle, used in [12]

Case Study

Control of an autonomous vehicle, used in [12]

- 4×4 manycore chip with a mesh-NoC

Case Study

Control of an autonomous vehicle, used in [12]

- 4×4 manycore chip with a mesh-NoC
- 33 tasks

Case Study

Control of an autonomous vehicle, used in [12]

- 4×4 manycore chip with a mesh-NoC
- 33 tasks
- 38 data flows

Case Study

Control of an autonomous vehicle, used in [12]

- 4×4 manycore chip with a mesh-NoC
- 33 tasks
- 38 data flows
- 4 **non-shared virtual channels** (configuration-specific VC mapping)

Case Study

Control of an autonomous vehicle, used in [12]

- 4 $\times$ 4 manycore chip with a mesh-NoC
- 33 tasks
- 38 data flows
- 4 **non-shared virtual channels** (configuration-specific VC mapping)

Analysis

Computations for **different buffer sizes** (2, 100 flits and infinite)

Case Study

Control of an autonomous vehicle, used in [12]

- 4 × 4 manycore chip with a mesh-NoC
- 33 tasks
- 38 data flows
- 4 **non-shared virtual channels** (configuration-specific VC mapping)

Analysis

Computations for **different buffer sizes** (2, 100 flits and infinite)
Comparison with state-of-the-art approach [12] based on ST

Case Study

Control of an autonomous vehicle, used in [12]

- 4 $\times$ 4 manycore chip with a mesh-NoC
- 33 tasks
- 38 data flows
- 4 **non-shared virtual channels** (configuration-specific VC mapping)

Analysis

Computations for **different buffer sizes** (2, 100 flits and infinite)
Comparison with state-of-the-art approach [12] based on ST
Further computations with **shared VCs**

Comparative Results

Delay bounds and simulated delays

	$B = 2$	$B = 100$	$B = \infty$
Average tightness	64%	67%	71%
Average tightness difference	+0.07%	+0.08%	-0.03%
Maximum tightness difference	+3.70%	+3.49%	+0.01%
Minimum tightness difference	-0.10%	-0.10%	-0.10%

Comparative Results

Delay bounds and simulated delays

	$B = 2$	$B = 100$	$B = \infty$
Average tightness	64%	67%	71%
Average tightness difference	+0.07%	+0.08%	-0.03%
Maximum tightness difference	+3.70%	+3.49%	+0.01%
Minimum tightness difference	-0.10%	-0.10%	-0.10%

- Delay bounds are **very similar** to [12]

Comparative Results

Delay bounds and simulated delays

	$B = 2$	$B = 100$	$B = \infty$
Average tightness	64%	67%	71%
Average tightness difference	+0.07%	+0.08%	-0.03%
Maximum tightness difference	+3.70%	+3.49%	+0.01%
Minimum tightness difference	-0.10%	-0.10%	-0.10%

- Delay bounds are **very similar** to [12]
- On average, they differ by less than 0.1%

Comparative Results

Delay bounds and simulated delays

	$B = 2$	$B = 100$	$B = \infty$
Average tightness	64%	67%	71%
Average tightness difference	+0.07%	+0.08%	-0.03%
Maximum tightness difference	+3.70%	+3.49%	+0.01%
Minimum tightness difference	-0.10%	-0.10%	-0.10%

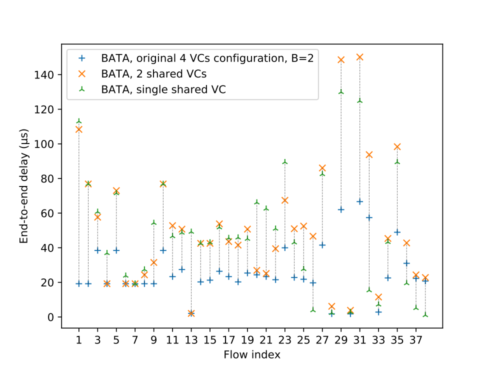
- Delay bounds are **very similar** to [12]
- On average, they differ by less than 0.1%
- **Good tightness**: 64% to 71% on average

Extended Results

We reduced the number of VCs and computed the WC bounds

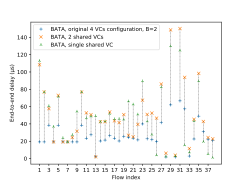
Extended Results

We reduced the number of VCs and computed the WC bounds



Extended Results

We reduced the number of VCs and computed the WC bounds

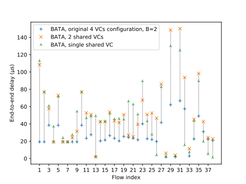


Improvements of BATA

- All flows remain schedulable with 2 shared VCs

Extended Results

We reduced the number of VCs and computed the WC bounds

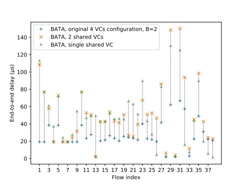


Improvements of BATA

- All flows remain schedulable with 2 shared VCs
- All flows remain schedulable with one shared VC

Extended Results

We reduced the number of VCs and computed the WC bounds

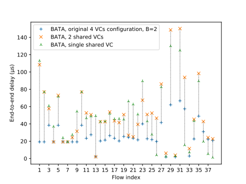


Improvements of BATA

- All flows remain schedulable with 2 shared VCs
- All flows remain schedulable with one shared VC
- Wider applicability domain

Extended Results

We reduced the number of VCs and computed the WC bounds



Improvements of BATA

- All flows remain schedulable with 2 shared VCs
- All flows remain schedulable with one shared VC
- Wider applicability domain
- Computation times with shared VCs are higher but remain reasonable

Summary

BATA is:

- **generic** (multiple shared VCs, priority-sharing, buffer size. . .)

Summary

BATA is:

- **generic** (multiple shared VCs, priority-sharing, buffer size. . .)
- **tight**

Summary

BATA is:

- **generic** (multiple shared VCs, priority-sharing, buffer size. . .)
- **tight**
- . . . but **computationally expensive**!

Perspectives

- **Computational aspect**: reduce complexity of the approach

Perspectives

- **Computational aspect**: reduce complexity of the approach
- **Applicability**: extend model to support bursty traffic on heterogeneous platforms

Perspectives

- **Computational aspect**: reduce complexity of the approach
- **Applicability**: extend model to support bursty traffic on heterogeneous platforms
- **Integration** : integrate model in DSE methodologies

Thank you for your attention!

- Z. Shi and A. Burns, "Real-time communication analysis with a priority share policy in on-chip networks," in *21st Euromicro Conference on Real-Time Systems*, pp. 3–12, July 2009.
- Q. Xiong, F. Wu, Z. Lu, and C. Xie, "Extending real-time analysis for wormhole nocs," *IEEE Transactions on Computers*, vol. PP, no. 99, pp. 1–1, 2017.
- M. Liu, M. Becker, M. Behnam, and T. Nolte, "Tighter time analysis for real-time traffic in on-chip networks with shared priorities," in *10th IEEE/ACM International Symposium on Networks-on-Chip*, 2016.
- S. Tobuschat and R. Ernst, "Real-time communication analysis for networks-on-chip with backpressure," in *Design, Automation Test in Europe Conference Exhibition*, 2017.

- E. A. Rambo and R. Ernst, "Worst-case communication time analysis of networks-on-chip with shared virtual channels," in *Proceedings of Design, Automation Test in Europe Conference Exhibition*, 2015.
- T. Ferrandiz, F. Frances, and C. Fraboul, "A method of computation for worst-case delay analysis on spacewire networks."
- L. Abdallah, M. Jan, J. Ermont, and C. Fraboul, "Wormhole networks properties and their use for optimizing worst case delay analysis of many-cores," in *10th IEEE International Symposium on Industrial Embedded Systems (SIES)*, pp. 1–10, June 2015.

- Y. Qian, Z. Lu, and W. Dou, "Analysis of worst-case delay bounds for best-effort communication in wormhole networks on chip," in *Networks-on-Chip, 3rd ACM/IEEE International Symposium on*, May 2009.
- F. Jafari, Z. Lu, and A. Jantsch, "Least upper delay bound for vbr flows in networks-on-chip with virtual channels," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 20, pp. 35:1–35:33, June 2015.
- A. Mifdaoui and H. Ayed, "Buffer-aware worst case timing analysis of wormhole network on chip," *arXiv*, vol. abs/1602.01732, 2016.
- F. Giroudot and A. Mifdaoui, "Buffer-aware worst-case timing analysis of wormhole nocs using network calculus," in *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, (Porto, PT), pp. 1–12, 2018.

B. Nikolic, S. Tobuschat, L. Soares Indrusiak, R. Ernst, and A. Burns, "Real-time analysis of priority-preemptive nocs with arbitrary buffer sizes and router delays," *Real-Time Systems*, 06 2018.

M. Liu, M. Becker, M. Behnam, and T. Nolte, "Buffer-aware analysis for worst-case traversal time of real-time traffic over rra-based nocs," in *27th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, March 2017.

M. Boyer, B. Dupont De Dinechin, A. Graillat, and L. Havet, "Computing Routes and Delay Bounds for the Network-on-Chip of the Kalray MPPA2 Processor," in *ERTS 2018 - 9th European Congress on Embedded Real Time Software and Systems*, (Toulouse, France), Jan. 2018.

V. Catania, A. Mineo, S. Monteleone, M. Palesi, and D. Patti, "Cycle-accurate network on chip simulation with noxim," *ACM Trans. Model. Comput. Simul.*, vol. 27, pp. 4:1–4:25, Aug. 2016.